数据可视化 Final Project

赵海凯 16307110258
王博俨 14307130045
钟祖远 14307130216

1. 项目介绍

在本项目中，我们实现了对人脸图片数据的三种处理：人脸互换（face swap）、人脸融合（face morph）以及基于特征向量的人脸处理（eigen face）。

1.1. 人脸互换（face swap）

人脸互换部分主要实现的功能是，给定任意两张人脸图片，通过一系列操作，使两个人的脸部交换，这部分需要的问题有：

1. 不同的人的脸部结构千差万别，同一个人也会因为角度、面部表情的不同而导致差别，即如何实现不同图片的人脸对齐；
2. 不同人脸的肤色、光照不同，即不同图片的面部亮度不同，在换脸后如何与整体亮度统一；
3. 不同人脸的纹理不同，比如老人的皱纹等，如何实现换脸后纹理的统一。

1.2 人脸融合（face morph）

在人脸融合部分，我们需要实现给定任意两张人脸图片和融合度α，通过一系列操作，实现两个人脸的融合。这一部分的困难在于：

1. 人脸结构的检测与分割。对于给定的人脸图片，人脸的结构差异很大。
2. 人脸融合度的构建。对于给定的融合度 α，如何对两张图片的人脸取样与映射。

1.3 本征脸（eigen face）

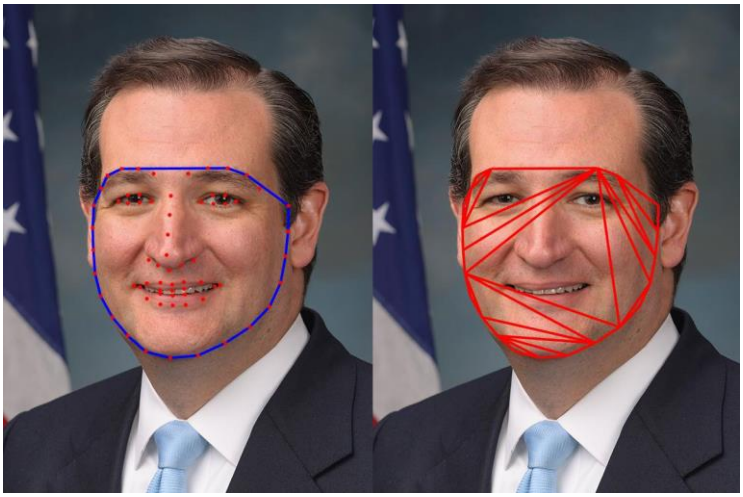
在这一部分，需要对较大的数据集（几百张，几千张人脸图片）进行处理，通过主成分分析的方法，得到一定数量的人脸主成分。这一部分的主要困难在于数据集的预处理，我们需要将不同图片中的人脸对齐，才能进行后续的处理。

2. 算法结构与处理过程

2.1 人脸变换

2.1.1 人脸关键点检测

实现人脸变换的第一步，便是人脸关键点的检测，得到图片中的人脸的结构。在这里，我们利用旷视 face++的 API 来定位人脸关键的集合{V1,V2,⋯,Vn}，其中关键点的数量可以选择 83 或者 106.



图一：左：人脸关键点与凸包；右：德劳内三角集

2.1.2 计算凸包

在获取人脸关键点集合后，我们需要计算这些关键点的凸包（convex hull）（凸包是一个计算几何（图形学）中的概念：在一个实数向量空间 V 中，对于给定集合 X，所有包含 X 的凸集的交集 S 被称为 X 的凸包。X 的凸包可以用 X 内所有点(X1，...Xn)的凸组合来构造.）在这里，我们计算凸包是为了获取这些人脸关键点组成的一个人脸区域。

2.1.3 德劳内（Delaunay）三角划分

在获得凸包以后，我们对凸包内的人脸关键点进行德劳内三角的划分。德劳内三角划分能将我们的凸包区域进行分割，并且更易于保留人脸的细节部分，并且因为获取仿射变换需要原图片和目标图片的各三个点，正好对应于原图和目标图的对应的德劳内三角。

2.1.4 进行仿射变换

在获得原图片和目标图图片的德劳内三角以后，我们需要寻找两张图对应的三角形对，对这样的每一对三角形，我们可以计算得到一个仿射函数。这个仿射函数将被用于三角形对之间的仿射变换。重复这个操作，直到所有区域都操作完毕，我们得到了人脸的位置变换。

2.1.5 无缝融合

在上述人脸仿射变换后，我们得到人脸结构和位置的变换，但我们没有对人脸区域亮度进行调整，这样会造成人脸区域和其他区域的颜色协调的问题。所以最后我们用 opencv 的无缝融合函数 seamlessClone()来实现无缝融合操作。

2.2 人脸融合

2.2.1 人脸关键点检测

和 2.1.1 部分一样，我们首先需要确定给定图片的人脸的结构，所以需要检测人脸的关

键点。在这里我们仍然使用旷视 face++的人脸关键点识别 API，选择 108 点检测。

2.2.2 定义融合度

要实现两张人脸的融合，我们还要定义融合度 α（0=<α<=1）：

$$M(x,y)=(1-\alpha)I(x,y)+\alpha J(x,y) \quad \begin{matrix}\alpha\to 0, result\to I \\ \alpha\to 1, result\to J\end{matrix}$$

M 是融合后的图像，I，J 是两张待融合图像。我们定义 α，使 α 趋于 0 使，越来越接近图像 I；α 趋于 1 时，越来越接近图像 J；α 等于 0.5 时，相当于两张图像的平均。

2.2.3 采样点加权

在获得两张人脸的关键点后，我们再对两张人脸进行加权平均：

$$x_m=(1-\alpha)x_i+\alpha x_j$$

$$y_m=(1-\alpha)y_i+\alpha y_j$$

通过这个式子，我们计算加权后的人脸关键点的位置。

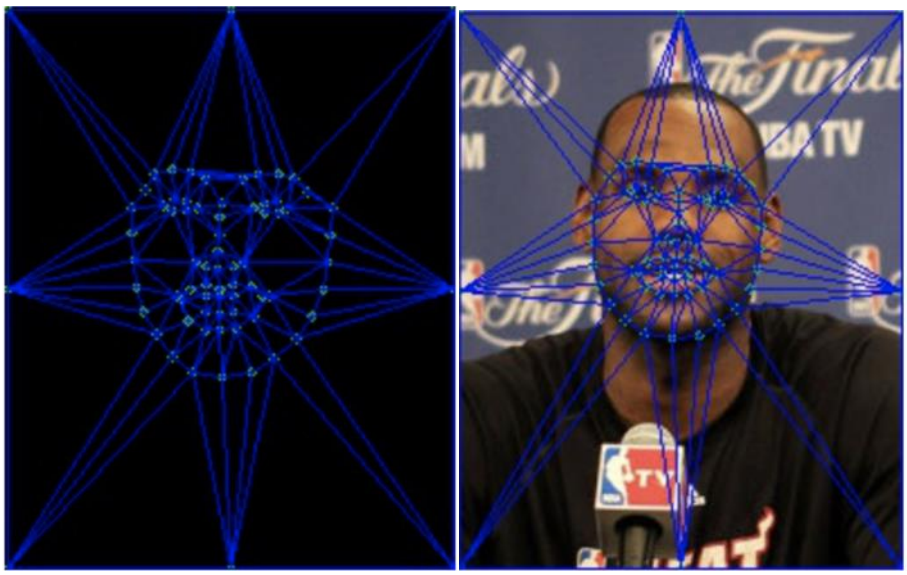


图 2：左：采样点加权后的人脸结构；右：德劳内三角划分

2.2.4 德劳内三角划分

如同 1.1.3，我们对获取的人脸关键点进行德劳内三角划分，不同的是，我们为了获取人脸框架结构在整张图中的位置，需要手动添加图片四个角以及四边中心点作为辅助点，再进行德劳内三角划分。这样，我们就得到非常详细的人脸结构。

2.2.5 图像融合

在经过以上步骤后，我们进行人脸融合。首先，我们对源图像的人脸关键点对我们在 2.2.3 中得到的加权后的人脸关键点进行仿射变换。由对应的德劳内三角形确定的三个点，我们可以确定一个仿射变换；对所有的三角形进行这样的操作，我们就得到了仿射后的人脸图片。对两张源图进行这样的操作，我们就得到两个仿射后的人脸图片，再运用我们定义的融合度 α 进行加权平均，最终得到我们的目标图片。

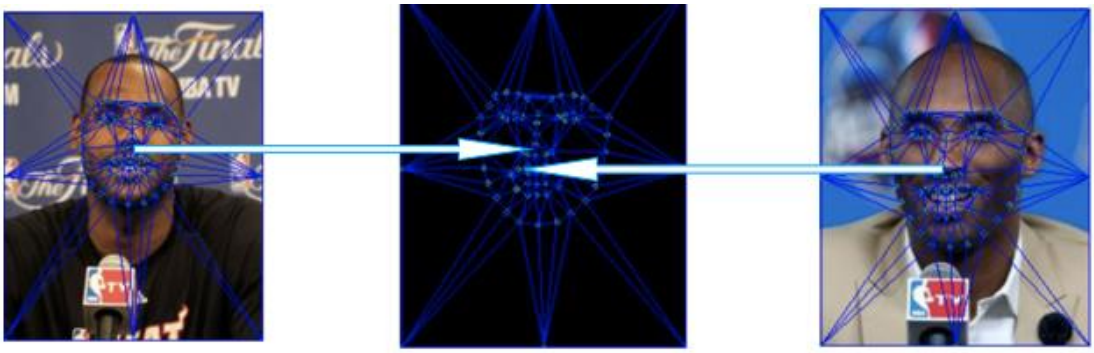


图 3：人脸融合

2.3 本征脸

2.3.1 数据预处理

此次我们用于测试的数据集有两个，一个是 NBA 现役球员，另一个是女外女明星。对于我们的人脸图片数据集，我们首先要做的是将它们的大小固定为统一的大小。然后，我们需要将每张人脸配准定位，一个方法是将每张图片的人的眼睛定位于同一水平线上，并且使每张图片的两眼中心点在相同的位置。

2.3.2 主成分分析

在预处理图片后，我们对图片的数据进行主成分分析。首先，对于每张 NxN 大小的图片，我们将其转化成一个（NxNx3）x 1 的向量。当有 M 张图片时，我们便有一个（NxNx3）x M 的矩阵。

在计算主成分之前，我们先来对我们的图片矩阵数据求平均值向量，这是为了后面的方差和协方差准备的。在获取了平均值向量之后，我们来计算主成分。在数数学上，我们通过线性代数中的矩阵特征值分解，可以找到矩阵的特征值和相应的特征向量，我们将这些特征值按从大到小排列，越大的特征值说明数据在这个特征值对应的特征向量的方向上的方差越大，这就是我们要找的主成分。在这里，我们可以设置参数，来选择主成分的数量。例如，我们选择前十大的特征值和对应的特征向量。

2.3.3 获得本征脸

在获得主成分之后，我们将选取的主成分矩阵，按原来图片矩阵转化为向量的逆操作，将矩阵变成图片矩阵。

3. 代码结构

3.1 人脸互换

相关文件：faceswap.py

1. get_points 函数。用于封装好 Face++的 API，得到图片的 83 个人脸特征点，并存储在一个列表当中，方便后续处理。
2. AffineTransform 函数。该函数输入源图、源图的德劳内三角、目标图的德劳内三角和给定的大小。通过计算获取仿射变换函数，并且进行仿射变换，输出变换后的图片。
3. DelaunayTriangles 函数。用于计算我们人脸关键点的德劳内三角。我们应用 openCV 的

Subdiv2D 函数，可以得到我们的德劳内三角。

4. warpTriangle 函数。通过德劳内三角的划分，我们定义这个函数，用来实现对德劳内三角的仿射变换，并且最终将左右变换后的三角形区域合并，得到我们的目标脸。
5. 最后是主函数，调用这些定义好的函数，读取我们的图片进行处理。

3.2 人脸融合

有关文件：morph.py

1. detect 函数。用于封装好 Face++的 API，得到图片的 108 个人脸特征点，并存储在一个字典当中，方便后续处理。
2. addBorderPoints 函数。该函数功能是给处理图片准备要划分三角形的时候，把图片四个顶点和四条边的中点的坐标添加到带划分的点中，这让得劳内三角形能完整的划分整张图片。
3. delaunaryTriangles 函数。由特征点的坐标生成其用来划分这些特征点的得劳内三角形。
4. affineTransform 函数。根据划分好的三角形，我们可以用来计算一张图的三角形到另外一张图所对应的三角形的仿射变换，该函数在 morphingTriangle 函数中使用。
5. morphingTriangle 函数。对两个仿射后的人脸图片，运用我们定义的融合度 α 进行加权平均，最终得到我们的目标图片。
6. morphing 函数。总的融合函数的接口，得到融合图片的脸部特征点的坐标,与新图片的脸部划分三角形坐标，调用 morphingTriangle 函数得到新的融合图片。
7. getPic 函数。 用于实现 GUI 中根据 alpha 生成一张图片的功能
8. get20Pics，create，sortKey,createGIF 函数。用于实现 GUI 生成一张 gif 图片的功能。

3.3 本征脸人脸

相关文件：eigenface_origin.py

1. readImages 函数。从指定的文件夹中读取图片文件，将其转化成 numpy 矩阵，放到一个 list 中。在此过程中，每张图片翻转后又保存了一次。
2. createdatamatrix 函数。将保存图片的 numpy 矩阵压平。(每张图片压平)
3. createnewface 函数。产生特征脸并且用 opencv 可视化界面展示。
4. resetslidervalues 函数。在 opencv 可视化界面中，点击一下图片，就重置所有 bar。
5. 总体思路：从文件夹中读取图片，变成矩阵，每张图片 flatten，用求平均向量，特征向量，再计算出 eigenface。

3.4 GUI 部分

有关文件: mainPage.py 主界面 GUI
morphGUI.py 融合脸功能界面 GUI
swapGUI.py 换脸功能界面 GUI
eigenGUI.py 特征脸界面 GUI

4. 开发环境

在本次项目中，算法几接口 API 都是由 pyhton3.5（python3.6）完成的，其中还使用了

openCV 库。GUI 使用 pyqt 完成。API 使用 face++人脸关键点接口。

5. 可执行文件及使用的数据集

5.1 数据集

我们有 2 个数据集，其中一个 image_swap 是用于测试换脸操作的；另外一个 image_eigen 数据集是用来测试本征脸的，是 NBA 现役球员的图片。

人脸融合测试图片，一定要是放在与该 py 文件夹中相同的文件夹，所以没有专门的数据集，可以从其他地方复制一些图片，进行测试。

5.2 可执行文件及运行方式

在 windows 下，我们有可执行文件 face#.exe。

执行方式：双击打开，会看到我们 GUI 的初始界面。在界面中我们会看到三个按钮，这是我们 GUI 实现的三个功能，及人脸互换、人脸融合和本征脸。点击其中一个按钮，会进入下一层界面，具体的操作可以参考我们附加提交的操作演示视频。

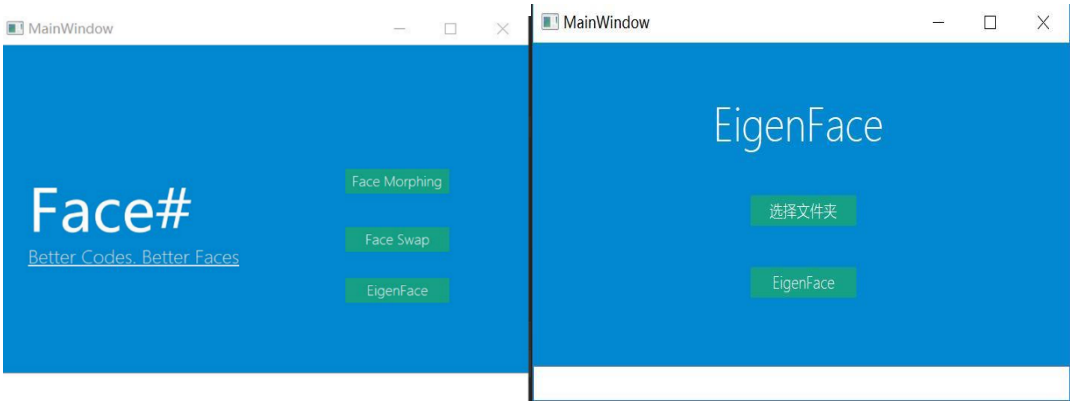


图 4：左：GUI 初始界面；右：本征脸界面

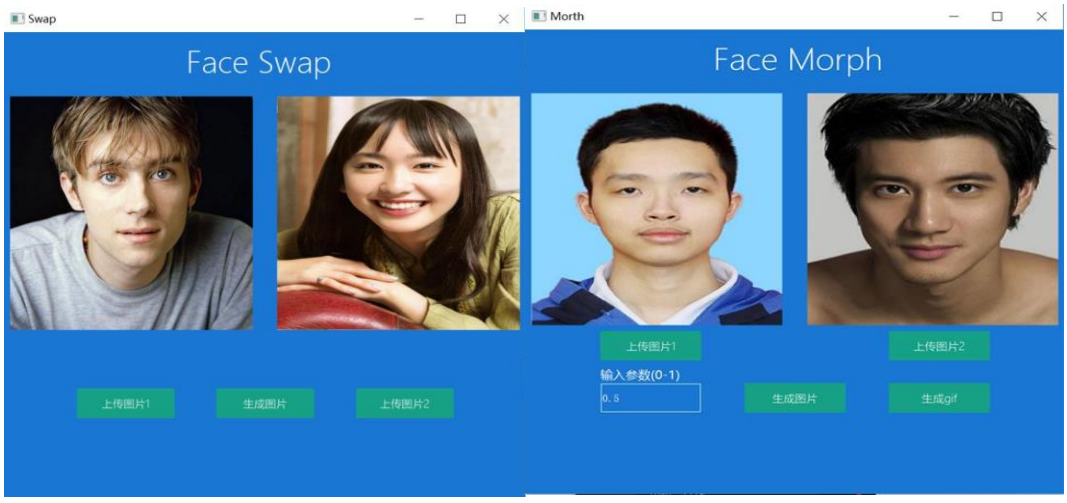


图 5：左：人脸互换界面；右：人脸融合界面

在 macOS 系统下，可以通过 python 文件名.py 命令运行我们的 python 文件。

6. 项目成果与反思

6.1 项目成果

6.1.1 人脸互换

图 6：人脸互换：左：源图，背景；中：结果图；右：源图，前景

从测试结果图中，我们可以看到，我们的人脸互换效果还是非常好的：从前景图上截取的脸部，能够非常好地与背景图融合；在边缘部分，没有明显的边界；在人脸的姿势上，也根据背景的姿势得到了调整，能够与背景姿势相一致。

当然我们也可以看到，在第二个结果中，由于前景脸和背景脸的脸部颜色相差太大，导致结果图中的脸部与额头等地方有明显的不一致，这是因为我们在做人脸互换的时候，只根据人脸关键点确定的凸包进行变换，而人脸关键点是没有包含额头及脖子部分的。这是我们人脸互换做的不好的地方

6.1.2 人脸融合

图 7：人脸融合：左：源图；中：结果图；右：源图

同样，对于我们人脸融合的结果，也非常好。在它融合度为 0.5 的情况下，包含了两张源图中一样多的特征，让我们能在结果图中能找到明显的属于两张源图的特征。

但是同样存在一些问题，比如在上面一张结果中，我们可以看到头发部分有重影。这是因为两张源图片的头发差异部分存在白色的背景，当进行融合时，由于有两部分的特征，所以会造成重影；但是第二张结果图中，我们没有看到这个现象，这是因为两张源图的差异部分没有很白的背景颜色。

6.1.3 本征脸

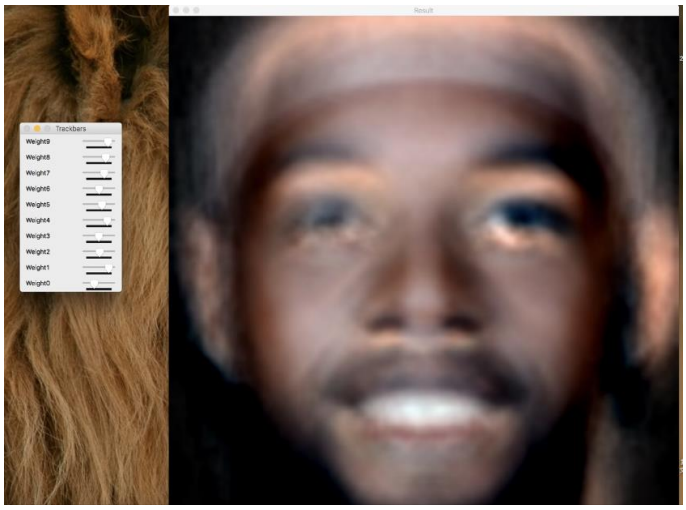


图 8：NBA 球员本征脸：左侧是主成分调节拖动条，右侧是结果

对于我们的本征脸，我们在此解释一下左侧拖条的含义：

首先我们这部分是对人脸数据集进行主成分分析，所谓的主成分就是数据中方差最大的一些方向，对应到人脸数据中，便是影响数据集中人脸的形态的最大的方向，或者说数据集中人脸变化最大的一些特征。比如，NBA 球员数据集中，球员的肤色有黑有白，这是变化非常大的特征，所以应该会有一个主成分只指向球员肤色的黑白的。

而之所以右侧的结果会很模糊，正是因为我们计算出的主成分代表了人脸数据集中变化最大的特征，所以这些特征组成的图片必定是非常模糊非常不整齐的。

6.2 项目思考与改进

6.1.1 人脸互换

我们的人脸互换算法依赖于人脸关键点锁定的区域，基本上限定于眉毛-两侧脸颊-下巴组成的区域，当两张源图的脸部颜色差别很大时，会出现结果图中脸部与额头差别很大的问题。所以，我们设想后续的优化是，将脸部的区域扩大到额头，乃至整个脖子，但是由于一般人脸检测的算法只限于原先的区域，如何检测这些扩展的区域，是一个较难解决的问题。

6.1.1 人脸融合

对于人脸融合中出现的结果图中脸部区域以外的重影问题，我们设想的优化是，将脸部以外的区域，全部设置成其中一张源图的背景区域。

6.2.3 本征脸

对于主成分分析得出来的若干个对人脸变化影响最大的特征，其中一些我们能够直观地看出来，例如肤色的黑白等，但有一些特征并不能很容易地得知与什么相关，需要我们进一步测试、观察与分析。

7. 小组成员工作内容

赵海凯：人脸融合的实现，GUI 的实现，视频展示
王博俨：本征脸的实现，ppt 制作，讲演
钟祖远：人脸互换的实现，视频剪辑，报告撰写